APPLICATION

FOR

UNITED STATES LETTERS PATENT

APPLICANT NAME: Kent F. Hayes, Jr.

TITLE: METHOD, SYSTEM AND PROGRAM PRODUCT FOR CONTROLLING NATIVE APPLICATIONS USING OPEN SERVICE GATEWAY INITIATIVE (OSGi) BUNDLES

DOCKET NO.: RSW920030231US1

INTERNATIONAL BUSINESS MACHINES CORPORATION

CERTIFICATE OF MAILING UNDER 37 CFR 1.10

I hereby certify that, on the date shown below, this correspondence is being deposited with the United States Postal Service in an envelope addressed to Mail Stop Patent Application, Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450 as "Express Mail Post Office to Addressee" Mailing Label No. EV393299574US

on February 26, 2004

Jennifer Desbiens
Name of person mailing paper

Jennifer Desbiens 02/26/2004
Signature Date

METHOD, SYSTEM AND PROGRAM PRODUCT FOR CONTROLLING NATIVE APPLICATIONS USING OPEN SERVICE GATEWAY INITIATIVE (OSGi) BUNDLES

Background of the Invention

1. Field of the Invention

[0001] In general, the present invention generally relates to a method, system and program product for controlling native applications using open service gateway initiative (OSGi) bundles. Specifically, the present invention allows native applications running in a native environment on a client device to be controlled through OSGi bundles running in an OSGi environment on the client device.

2. Related Art

[0002] As computer networking has become more advanced, a standard known as the Open Service Gateway Initiative (OSGi) has been developed. The OSGi is an industry plan to provide a standard way to deliver managed services to devices and local networks. With such a standard, home users could, for example, change the setting on their thermostat from a remote location (e.g., the workplace). In general, the OSGi provides a good framework for developing application components. Under the OSGi, a basic component is known as an OSGi bundle. An OSGi application can be made up of combinations/suites of bundles that might use common functionality. To this extent, the OSGi allows developers to define the dependencies between the bundles such as the packages and services required by the bundles. The OSGi runtime can also determine whether a device has the necessary packages and resources. In a typical

implementation, an OSGi architecture will include, among other components, a server and one or more client devices. Each client device will have an OSGi environment within which OSGi applications are deployed. Using a management program on the server, the functions of the OSGi applications can be controlled. One type of function often performed on OSGi client devices is life cycle management. Life cycle management allows OSGi applications to be started, stopped, updated, installed or uninstalled from the server.

[0003] Unfortunately, as convenient as the OSGi framework can be, it fails to provide for similar management/control of native applications on the client device. For example, a standard desktop (e.g., WIN-32) computer could have several native application that run within a native environment. Typical examples of native applications include word processing programs, spreadsheets, etc. Since such applications are widely used, it would be highly advantageous to be able to control them in a similar manner from the OSGi environment. Currently, the only way to control the functions of the native application from the server is to write separate programs for each desired function. Not only is this extremely tedious, but it is also highly inefficient.

[0004] In view of the foregoing, there exists a need for a method, system and program product for controlling native applications using OSGi bundles. Specifically, a need exists for a way to enable life cycle management of native applications using OSGi bundles.

Summary of the Invention

[0005] In general, the present invention provides a method, system and program product for controlling (e.g., managing a life cycle of) native applications using OSGi bundles. Specifically, under then present invention, a native application is packaged within an OSGi bundle to create a

link therebetween. Information describing the commands needed to control the life cycle of the bundle (install, uninstall, start, stop) may also be stored within the OSGi bundle during this packaging step. In any event, the packaged OSGi bundle is installed within an OSGi environment of a client device. Once installed, the packaged OSGi bundle is deployed in a native environment of the client device and the native application is optionally removed from within the packaged OSGi bundle while maintaining the link. Thereafter, the native application within the native environment can be controlled from the server using the OSGi bundle within the OSGi environment.

[0006] A first aspect of the present invention provides a method for controlling native applications using Open Service Gateway Initiative (OSGi) bundles, comprising: packaging a native application within an OSGi bundle to create a link between the OSGi bundle and the native application (information describing the commands needed to control the life cycle of the bundle (install, uninstall, start, stop) may be stored within the bundle during the packaging step); installing the OSGi bundle within an OSGi environment of a client device after the packaging; deploying the OSGi bundle within a native environment of the client device; and controlling the native application within the native environment using the OSGi bundle within the OSGi environment.

[0007] A second aspect of the present invention provides a method for enabling life cycle management of native applications using Open Service Gateway Initiative (OSGi) bundles, comprising: packaging a native application within an OSGi bundle on a server to create a link between the OSGi bundle and the native application (information describing the commands needed to control the life cycle of the bundle (install, uninstall, start, stop) may be stored within

the bundle during the packaging step); installing the OSGi bundle within an OSGi environment of a client device after the packaging; deploying the OSGi bundle within a native environment of the client device; removing the native application from within the OSGi bundle while maintaining the link; and managing a life cycle of the native application within the native environment using the OSGi bundle in the OSGi environment.

[0008] A third aspect of the present invention provides a system for controlling native applications using Open Service Gateway Initiative (OSGi) bundles, comprising: a packaging system for packaging a native application within an OSGi bundle to create a link between the OSGi bundle and the native application (information describing the commands needed to control the life cycle of the bundle (install, uninstall, start, stop) may be stored within the bundle during the packaging step); an exportation system for installing the OSGi bundle within an OSGi environment of a client device, wherein the OSGi bundle is thereafter deployed within a native environment of the client device; and a control system for controlling the native application within the native environment using the OSGi bundle within the OSGi environment.

[0009] A fourth aspect of the present invention provides a system for controlling native applications using Open Service Gateway Initiative (OSGi) bundles, comprising: means for packaging a native application within an OSGi bundle to create a link between the OSGi bundle and the native application; means for installing the OSGi bundle within an OSGi environment of a client device; means for deploying the OSGi bundle within a native environment of the client device; means for removing the native application from within the OSGi bundle while maintaining the link; and means for managing a life cycle of the native application within the native environment using the OSGi bundle within the OSGi environment.

[0010] A fifth aspect of the present invention provides a program product stored on a recordable medium for controlling native applications using Open Service Gateway Initiative (OSGi) bundles, which when executed, comprises: program code for packaging a native application within an OSGi bundle to create a link between the OSGi bundle and the native application (information describing the commands needed to control the life cycle of the bundle (install, uninstall, start, stop) may be stored within the bundle during the packaging step); program code for installing the OSGi bundle within an OSGi environment of a client device, wherein the OSGi bundle is thereafter deployed within a native environment of the client device; and program code for controlling the native application within the native environment using the OSGi bundle within the OSGi environment.

[0011] Therefore, the present invention provides a method, system and program product for controlling native application using OSGi bundles.

Brief Description of the Drawings

[0012] These and other features of this invention will be more readily understood from the following detailed description of the various aspects of the invention taken in conjunction with the accompanying drawings in which:

[0013] Fig. 1 depicts an illustrative system for controlling native applications using OSGi bundles according to the present invention.

[0014] Fig. 2 depicts the system of Fig. 1 in greater detail.

[0015] Fig. 3 depicts a method flow diagram according to the present invention.

[0016] It is noted that the drawings of the invention are not necessarily to scale. The drawings are merely schematic representations, not intended to portray specific parameters of the invention. The drawings are intended to depict only typical embodiments of the invention, and therefore should not be considered as limiting the scope of the invention. In the drawings, like numbering represents like elements.

Detailed Description of the Drawings

[0017] For convenience purposes, the Detailed Description of the Drawings will have the following sections:

I. General Description

II. Detailed Example

I. General Description

[0018] As indicated above, the present invention provides a method, system and program product for controlling (e.g., managing a life cycle of) native applications using OSGi bundles. Specifically, under the present invention, a native application is packaged within an OSGi bundle to create a link therebetween. In packaging the native application within the OSGi bundle, information describing the commands needed to control the life cycle of the bundle (install, uninstall, start, stop) may be stored within the OSGi bundle as well. In any event, the OSGi bundle is installed within an OSGi environment of a client device. Once installed, the OSGi bundle is deployed in a native environment of the client device and the native application is

removed from within the OSGi bundle while maintaining the link. Thereafter, the native application can be controlled from the server through the OSGi bundle.

[0019] Referring now to Fig. 1, an illustrative system 10 for controlling native applications using OSGi bundles according to the present invention is shown. As depicted, system 10 includes server 12 and client device 14 (a single client device is shown for illustrative purposes only). It should be understood that the architecture shown herein is illustrative only and will likely include other known components not shown. For example, a typical embodiment of the invention would likely include a device, OSGi agents (on client device 14), a device management server and one or more application servers. Moreover, it should be understood that a typical embodiment of the invention could include multiple servers 12 and a network dispatcher. In any event, client device 14 is intended to represent any type of computerized device capable of communicating over a network. For example, client device 14 could be a desktop computer (e.g., WIN-32-based), a hand held device, a set top box, a home appliance, a security system, etc. In any event, server 12 and client device 14 typically communicate over any type of network such as the Internet, a local area network (LAN), a wide area network (WAN), a virtual private network (VPN), etc. As such, communication between server 12 and client device 14 could occur via a direct hardwired connection (e.g., serial port), or via an addressable connection that may utilize any combination of wireline and/or wireless transmission methods. Moreover, conventional network connectivity, such as Token Ring, Ethernet, WiFi or other conventional communications standards could be used. Still yet, connectivity could be provided by conventional TCP/IP sockets-based protocol. In this instance, client device 14 could utilize an Internet service provider to establish connectivity to server 12.

[0020] Under the present invention, server 12 will be provided with one or more OSGi bundles 16 and a native application 18. As known, an OSGi bundle is essentially a .JAR file with certain characteristics which enable it to effectively interact with the OSGi framework. As such, OSGi bundle 16 has the ability to be controlled in a well defined manner. To enable control of native application 18 on client device 14, management program 20 will create a link between native application 18 and OSGi bundle 16. In a typical embodiment, this is accomplished by packaging the native application 18 within OSGi bundle 16. Once packaged, management program 20 will install the OSGi bundle 16 (which now includes native application 18) within OSGi environment 22 of client device 14. Once installed, the OSGi bundle 16 will be deployed in native environment 24 (e.g., a WIN-32 environment) and then native application 18 will be removed OSGi bundle 16 while maintaining the link therewith. At this point, only native application 18 will remain in native environment 24, while OSGi bundle 16 will remain in OSGi environment 22. This allows OSGi bundle 16 to be thought of as a "control bundle." Specifically, when control over native application 18 from server 12 is desired, management program 20 can be used to issue "commands." These commands will be received by agent 26 and communicated to the OSGi bundle 16 in OSGi environment 22. Thereafter, OSGi bundle 16 will instruct agent 26 (e.g., a WIN-32 agent) to carry out the command on native application 18 in native environment 24.

II. Detailed Example

[0021] Referring now to Fig. 2, a more detailed diagram of Fig. 1 is shown. As shown, server 12 generally comprises central processing unit (CPU) 30, memory 32, bus 34, input/output (I/O)

interfaces 36, external devices/resources 38 and storage unit 40. CPU 30 may comprise a single processing unit, or be distributed across one or more processing units in one or more locations, e.g., on a client and computer system. Memory 32 may comprise any known type of data storage and/or transmission media, including magnetic media, optical media, random access memory (RAM), read-only memory (ROM), a data cache, etc. Moreover, similar to CPU 30, memory 32 may reside at a single physical location, comprising one or more types of data storage, or be distributed across a plurality of physical systems in various forms.

[0022] I/O interfaces 36 may comprise any system for exchanging information to/from an external source. External devices/resources 38 may comprise any known type of external device, including speakers, a CRT, LCD screen, handheld device, keyboard, mouse, voice recognition system, speech output system, printer, monitor/display, facsimile, pager, etc. Bus 34 provides a communication link between each of the components in server 12 and likewise may comprise any known type of transmission link, including electrical, optical, wireless, etc.

[0023] Storage unit 40 can be any system (e.g., database) capable of providing storage for information under the present invention. Such information could include, for example, information about which native applications are packaged within which OSGi bundles etc. As such, storage unit 40 could include one or more storage devices, such as a magnetic disk drive or an optical disk drive. In another embodiment, storage unit 40 includes data distributed across, for example, a local area network (LAN), wide area network (WAN) or a storage area network (SAN) (not shown). Although not shown, additional components, such as cache memory, communication systems, system software, etc., may be incorporated into server In addition, it

should also be appreciated that although not shown, client device 14 would likely include computerized components similar to server 12.

[0024] Shown in memory 32 of server 12 is management program 42. Under the present invention, management program 42 can include parts or all of any OSGi management program now known or later developed. However, under the present invention, management program 42 is adapted (i.e., includes certain sub-systems) to provide for the control of native applications using OSGi bundles (a function not previously realized). Specifically, as shown, management program 42 includes packaging system 44, export system 46, deployment system 48, removal system 50 and control system 52. It should be understood that each of these systems includes program code/logic for carrying out the functions described herein. To this extent, the systems could be realized as plugins or the like. Moreover, as will be further described below certain systems such as deployment system 48 and removal system 50 could be optional.

[0025] Regardless, under the present invention, packaging system 44 will package native application 18 within OSGi bundle 16 to create a link there between. The linkage of OSGi bundle 16 to life cycle functions (like uninstall) of native application 18 could be done generically if the environment permits (e.g., WIN-32), or more specifically for native application 18 via running scripts, executable file, etc. In a typical embodiment, native application 18 and OSGi bundle 16 are assigned the same name once the packaging has occurred. Thus, for example, the packaged OSGi bundle 16 could be assigned the name of native application 18. In any event, a record of the linkage (e.g., packaging) will be maintained by server 14 in storage unit 40. This allows the correct OSGi bundle to later be identified when control over the native application 18 on client device 14 is desired.

[0026] Once native application 18 is packaged within OSGi bundle 16, it will be registered with server (e.g., in storage unit 40 and/or cache). The OSGi bundle 16 containing the native application 18 is then available for distribution to appropriate client devices such as client device 14. In a typical embodiment, a management action is initiated to deploy OSGi bundle 16 (and hence native application 18) to OSGi environment 22 of client device 14. Thereafter, the act of starting OSGi bundle 16 causes native application 18 to be extracted from OSGi bundle 16, and the "Install" program for native application 18 to be executed. Native application 18 is then removed from OSGi bundle 16 leaving only a "controller bundle" with it's associated links to the native application 18.

[0027] In an optional alternative embodiment, separate optional program code components could perform these functions. For example, export system 46 could install OSGi bundle 16 (having native application 18 therein) within OSGi environment 22 of client device 14. Thereafter, deployment system 48 could deploy OSGi bundle 16 within native environment 24. Once deployed, removal system 50 would remove native application 18 from within OSGi bundle 16, and then remove OSGi bundle 16 from native environment 24. At this point, native application 18 is deployed in native environment 24, while OSGi bundle is deployed within OSGi environment 22. It should be understood that each system within management program 42 need not be loaded on server 12. For example, deployment system 48 and/or removal system 50 could alternatively be loaded on client device 14 or packaged within OSGi bundle 16 with native application 18.

[0028] Regardless, if control of native application 18 is later desired, such control can be enabled using OSGi bundle 16. For example, assume an operator or the like (not shown) wishes to

manage a life cycle of native application 18 (e.g., start, stop, install or uninstall). The operator would use control system 52 to issue a life cycle command 28 requesting a certain action to OSGi bundle 16. Since a record has been maintained on server 12 corresponding to the link between native application 18 and OSGi bundle 16, command system 52 can identify the correct OSGi bundle 16. The command 28 would be received by the agent and relayed to the OSGi bundle 16, which would instruct OSGi bundle 16 to carry out the instruction therein. In a typical embodiment, agent 26 is a WIN-32 agent that interfaces with native environment to perform the requested action such as starting, stopping, installing or uninstalling native application 18.

[0029] It should be understood that other embodiments of controlling native application 18 are also possible. For example, the command could be received by agent 26, relayed to OSGi bundle 16, which would then request agent 26 to carry out the command 28. Alternatively, the command 28 could be received by agent 26, relayed to OSGi bundle 16, which would carry out the command 28 itself. Still yet, the command 28 could be received by OSGi bundle 16 bundle, which would carry out the command 28 (e.g., via agent code or calls to agent code packed within OSGi bundle 16).

[0030] It should be appreciated that client device 14 could maintain its own record of links between OSGi bundles and native applications. Although not necessary, this could provide some redundancy in the system. It should also be understood that the present invention can be realized in hardware, software, or a combination of hardware and software. Any kind of computer system(s) - or other apparatus adapted for carrying out the methods described herein - is suited. A typical combination of hardware and software could be a general purpose computer system with a computer program that, when loaded and executed, carries out the respective methods

described herein. Alternatively, a specific use computer, containing specialized hardware for carrying out one or more of the functional tasks of the invention, could be utilized. The present invention can also be embedded in a computer program product, which comprises all the respective features enabling the implementation of the methods described herein, and which - when loaded in a computer system - is able to carry out these methods. Computer program, software program, program, or software, in the present context mean any expression, in any language, code or notation, of a set of instructions intended to cause a system having an information processing capability to perform a particular function either directly or after either or both of the following: (a) conversion to another language, code or notation; and/or (b) reproduction in a different material form.

[0031] Referring now to Fig. 3, a method flow diagram 100 according to the present invention is shown. As depicted, first step S1 is to package a native application within an OSGi bundle to create a link between the native application and the OSGi bundle. Second step S2 is to install the OSGi bundle within an OSGi environment of a client device. Third step is to deploy the OSGi bundle within a native environment of the client device. Fourth step S4 is to remove the native application from within the OSGi bundle while maintaining the link. Fifth step S5 is to control the native application within the native environment using the OSGi bundle in the OSGi environment

[0032] The foregoing description of the preferred embodiments of this invention has been presented for purposes of illustration and description. It is not intended to be exhaustive or to limit the invention to the precise form disclosed, and obviously, many modifications and variations are possible. Such modifications and variations that may be apparent to a person

skilled in the art are intended to be included within the scope of this invention as defined by the accompanying claims.